

Context Driven Technique for Document Classification

*Upasana Pandey, @ S. Chakraverty, # Rahul Jain

*upasana1978@gmail.com

@apmahs@rediffmail.com

#rahul.jain@nsitonline.in

NSIT, SEC 3, Dwarka, New Delhi 110078, India

Abstract: In this paper we present an innovative hybrid Text Classification (TC) system that bridges the gap between statistical and context based techniques. Our algorithm harnesses contextual information at two stages. First it extracts a cohesive set of keywords for each category by using lexical references, implicit context as derived from LSA and word-vicinity driven semantics. And secondly, each document is represented by a set of context rich features whose values are derived by considering both lexical cohesion as well as the extent of coverage of salient concepts via lexical chaining. After keywords are extracted, a subset of the input documents is apportioned as training set. Its members are assigned categories based on their keyword representation. These labeled documents are used to train binary SVM classifiers, one for each category. The remaining documents are supplied to the trained classifiers in the form of their context-enhanced feature vectors. Each document is finally ascribed its appropriate category by an SVM classifier.

Keywords: Lexical references, Vicinity driven semantics, Lexical chaining.

I. INTRODUCTION

Text Classification (TC) is the task of inspecting input documents with the aim of assigning them categories from a predefined set. TC serves a wide range of applications such as classifying scientific and medical documents, organizing documents for query-based information dissemination, email folder management and spam filtering, topic-specific search engines and library / web document indexing [1].

The world-wide-web has witnessed a mind boggling growth of both volume as well as variety of text content, driving the need for sophisticated TC techniques. Over the years, numerous statistical approaches have been successfully devised. But increasingly, the so-called bag-of-words approach has reached a point of saturation where it can no longer handle the increasing complexities posed by polysemous words, multi-word expressions and closely related categories that require semantic analysis. Recently, researchers have illustrated the potential for applying *context-oriented* approaches to improve upon the quality of TC. This has resulted in a fruitful convergence of several fields such as Information Retrieval (IR), Natural Language Processing (NLP) and Machine Learning (ML). The aim is to derive the semantics encapsulated in groups of words and utilize it to improve classification accuracy.

This paper showcases a proposal which bridges the gap between statistical and context based approaches for TC. The

contextual information is harnessed at two stages. First, it is used to extract a meaningful and cohesive set of keywords for each input category. Secondly, it is used to refine the feature set representing the documents to be classified.

For the rest of the paper, section II presents a brief background of the field and the relevance of context based TC. Section III brings into perspective prior work in the area. Section IV presents our proposed context-enhanced TC model. In Section V, we compare the proposed TC scheme with current approaches and overview its implications and advantages. We conclude in Section VI.

II. BACKGROUND AND MOTIVATION

A TC system accepts two primary inputs; a set of categories and a set of documents to be classified. Most TC systems use supervised learning methods that entail training a classifier. Support Vector Machines (SVM) use kernel functions to transform the feature space into a higher dimensional space and have been found to be especially suitable for TC applications [2]. Their training process may use prior labeled documents which guide the classifier in tuning its parameters. In another approach, as illustrated in the Sleeping Experts algorithm [3], a classifier is trained dynamically during the process of classification using a subset of the input documents and its parameters are progressively refined. Another approach is to identify a set of training documents from among the input dataset and label them with keyword- matched categories.

A concern here is to derive a set of keywords for each category. While user-input keywords can be used, this is cumbersome and may be impractical. It is attractive to generate the keywords automatically. This motivates the application of contextual information to cull out meaningful keywords for each category.

The documents must be pre-processed and represented by a well-defined and cohesive set of features; the most time-consuming part of TC. The main aim of feature extraction is to reduce the large dimensionality of a document's feature space and represent it in concise manner within a meaningfully derived context space. Once encoded as a feature set, a trained classifier can be invoked to assign it a suitable category. A plethora of classification techniques are available and have been tapped by TC researchers. They include Bayesian classifiers [4], SVM [5], Decision trees [6], Ada Boost [7], RIPPER [3], fuzzy classifiers [8] and

growing seeded clusters based on Euclidean distance between data points [9].

Predominantly, Statistical approaches have been applied for feature extraction with highly acceptable performance of about 98%. These methods employ statistical metrics for feature evaluation, the most popular being Term Frequency-Inverse Document Frequency (TF-IDF), Chi-square and Information Gain[10]. Statistical techniques have been widely applied to web applications and harnessed to capacity. Their applicability in further improving the quality of TC seems to have reached a saturation point. They have some inherent limitations, being unable to deal with synonyms, polysemous words and multi-word expressions.

On the other hand, Semantic or context-based feature extraction offers a lot of scope to experiment on the relevancy among words in a document. Semantic features encapsulate the relationship between words and the concepts or the mental signs they represent. Context can be interpreted in a variety of intuitively appealing ways such as:

- Implicit correlation between words as expressed through Latent Semantic Analysis (LSA)
- Lexical cohesiveness as implied by synonyms, hypernyms, hyponyms, meronyms or as sparse matrices of ordered words as described in [3]
- Syntactic relation between words as reflected by Parts-Of-Speech (POS) phrases
- Shallow semantic relationships as expressed by basic questions such as WHO did WHAT to WHOM, WHERE [2]
- Enhancement of a word's importance based on the influence of other salient words in its vicinity [11]
- Domain based context features expressed by domain specific ontology [7].

With such a wide spectrum, different context-oriented features can be flexibly combined together to target different applications. The issue of real time performance can be handled through efficient learning algorithms and compact feature representation.

In this paper, we propose a scheme for incorporating lexical, referential as well as latent semantics to produce context enhanced text classification.

III. REVIEW OF CONTEXT BASED TC

Several authors have suggested the use of lexical cohesion in different schemes to aid the process of classification.

L. Barak *et al* have used a lexical reference scheme to expand upon a given category name with the aim of automatically deriving a set of keywords for the category [12]. A reference vector comprises lexical references derived from wordnet and wikipedia for each category and a context vector is derived from the LSA space between categories and input documents. Each vector is used to calculate two cosine similarity scores between categories and documents. These scores are multiplied to assign a final similarity score. The final bootstrapping step uses the documents that are identified for each category to train its classifier. With their combined reference plus context approach, the authors have reported improvement in precision with Reuters-10 corpus.

In [5], Diwakar *et al* have used lexical semantics such as synonyms, analogous words and metaphorical words/phrases to tag input documents before passing them to a statistical Bayesian classifier. After semantic pre-processing and normalization, the documents are fed to a Naïve Byes classifier. The authors have reported improvement in classification accuracy with semantic tagging. However processing is almost entirely manual.

In [7], the authors locate the longest multiword expression leading to a concept. They derive concepts, synonyms and generalized concepts by utilizing ontology. Since concepts are predominantly noun phrases, a Part Of Speech (POS) analyzer is used to reject unlikely concepts. The set of terms and their lexically derived concepts is used to train and test a Ada-boost classifier. Using the Reuters-21578 [14], OHSUMED [15] and FAODOC [16] corpora, the authors report that the combined feature set of terms and derived concepts yield noticeable improvements over the classic term stem representation.

In [17] the authors select a set of likely positive examples from an unlabeled dataset using co-training. They also improve the positive examples with semantic features. The expanded positive dataset as well as the unlabeled negative examples together train the SVM classifier, thus increasing its reliability. For the final classification, the TF-IDF of all the semantic features used in the positive data sets are evaluated in test documents. The author's experiments on Reuters-21578 [14] and Usenet articles [18] corpora reveal that using positive and unlabeled negative datasets combined with semantic features gives improved result.

Ernandes *et al* [11] propose a term weighting algorithm that recognizes words as social networks and enhances the default score of a word by using a context function that relates it to neighboring high scoring words. Their work has been suitably applied for single word QA as demonstrated for crossword solving.

IV. PROPOSED WORK

Our proposal taps the potential of lexical and contextual semantics at two tiers:

A. Keyword extraction:

Automatic keyword generation frees the user from the encumbrance of inputting them manually. Our scheme employs lexical references derived from wordnet[19], implicit context as derived from an LSA and vicinity based context as given the terms surrounding already known keywords. The three driving forces together yield a cohesive set of keywords. The set of extracted keywords for each category serves as a template for identifying representative documents for that category. These documents are utilized as supervisors to train an SVM classifier.

B. Document representation:

Once training documents are identified, these and the remaining unlabeled test documents must be represented by a compact feature set. As during the keyword extraction phase, we use lexical relationships to identify generic

concepts, synonyms and specialized versions of terms. Taking a further step of refinement, we use anaphoric referential phrases and strong conjunctions to identify the length of cohesiveness or *lexical chain* of each salient concept. These parameters are utilized to fine tune TF-IDF feature values. Only those features whose values cross a stipulated threshold finally represent the document.

The algorithm Context-driven Text Classification CTC, is described in the pseudo-code in figure 1.

Step1: Removal of stop words and stemming:

The first step is pre-processing of documents. Stop words, i.e. those words that have negligible semantic significance such as 'the' 'an' 'a' etc are removed. Weak conjunctions such as 'and' are removed but strong conjunctions 'therefore' and 'so' are retained so that their contribution to lexical chaining can be utilized. Next the system carries out stemming, viz. the process of reducing inflected words to a root word form. The entire set of documents is partitioned into one third training set and two third testing set.

Step 2: Keyword extraction:

Words that bear a strong semantic connection with category names are keywords. This concept can be extended to hypothesize that words strongly associated with identified keywords are themselves good candidates as keywords. We apply these principles to extract keywords from the following sources.

Keyword Set- $K_{lex}(c)$ from lexical references:

Words that are lexically related to a category c are collected from the wordnet resource. Taking cue from [12], we use only that sense that corresponds to the category's meaning. First synonyms are extracted. Then hypernyms of each term are transitively located to allow generalization. Hypernyms are collected up to a pre-specified level k so that term meanings are not obfuscated by excessive generalization. Next, hyponyms at immediate lower level are located. At the end of this process, the system is armed with an initial set of keywords that explicitly refer to category names.

Keyword Set- $K_{imc}(c)$ from implicit context:

Latent Semantic Analysis (LSA) is a well known technique to help identify the implicit contextual bonding between category names and input documents. LSA uses the Singular Value Decomposition (SVD) to cull out groups of words which have high vector similarity in the LSA space [20]. This step generates a new set of contextually related keywords $AK_{imc}(c)$ for each category c .

Augmented Keyword Sets from vicinity context:

Words in the vicinity of keywords weave a semantic thread that correlates with the category. LSA does club together word co-occurring anywhere in a document but does not highlight semantics inherently present in neighboring words. The idea is to locate those words surrounding keywords that have a high potential for becoming keywords themselves. Given a keyword set,

the algorithm carries out a syntactic parsing of the sentences containing keywords. Since concepts are usually represented by noun phrases, they are separated out. The number of instances of each vicinity related pair is counted. If this exceeds a pre-decided threshold τ_v , the keyword set is augmented with the new term. The two augmented keyword sets are now $AK_{lex}(c)$ and $AK_{imc}(c)$.

Step 3: Evaluate documents' representative index:

For each document d in the training set, the algorithm calculates two r -scores per category c : $\rho(AK_{lex}(c), d)$ and $\rho(AK_{imc}(c), d)$ that denote the extent to which the document is *representative* of the two keyword sets respectively. Their product gives the overall r -score: $\rho(c, d) = \rho(AK_{lex}(c), d) * \rho(AK_{imc}(c), d)$.

The cosine similarity, which gives the cosine of the angle between two vectors has been used in [12,22] to match documents with categories. However the keyword vectors being equally weighted, the cosine similarity will bias documents with equal number of instances of each keyword against others. Also, a document with more instances of a keyword may receive the same similarity score as a document with less instances of the keyword. An appropriate metric should reflect the fact that any document containing more keywords from a category is more representative of it and a document with more instances of each keyword deals with the category in greater detail. Given a keyword list $AK(c)$ and a document d we multiply two factors to derive its representative index $\rho(AK(c), d)$: (1) fraction of the total number of category keywords present in d (2) Ratio of the average numbers of keywords in d to the sum total of these averages for all documents. Thus:

$$\rho(AK(c), d) = \frac{|AK(d)|}{|AK(c)|} \cdot \frac{\sum_i TF-IDF(w_i \in AK(d))}{\sum_k \sum_i TF-IDF(w_i \in AK(d_k))}$$

Step 4: Assigning categories to documents:

The next step labels documents with categories. The document with the highest r -score for a category is assigned to it. Thereafter, all documents whose r -scores for that category are within a preset *category assignment factor* F of this maximum similarity score are also assigned to it.

Step 5: Document feature representation:

The TC system utilizes contextual information to derive the feature set values of documents in the following ways:

Generalization with lexical references:

As a document is browsed, new terms encountered are annotated with their synonyms, hypernyms for k -up levels and hyponyms k -down levels. A rescanning of the document replaces any of the lexical references encountered with their equivalent base words. Term counts are simultaneously updated. When all documents have been tagged, TF-IDF values are evaluated.

Creating Lexical chains:

Vertical search engines focus on a particular topic or text segment only. They need TC methods with greater depth of

search. This is in contrast with generic Web search engines. Towards this goal, we recognize that the extent of coverage of a concept is as important as its frequency of occurrence. In our framework, we have introduced an optional step that can be invoked for vertical categorization requirements.

Lexical chaining:

Links consecutive sentences that are related through repetition of a concept term or its lexical references (iteration), anaphoric references and strong conjunctions [21]. Besides iteration, resolved anaphoric references also lead to the next link in the chain. Strong conjunctions such as therefore or so serve as final links. When a gap of a few sentences occurs, it indicates termination of a local chain.

The length of a chain indicates the extent of influence of a concept. The minimum length of any chain is a single sentence, as may be the case for most terms in a document. The TF-IDF values are weighted by their normalized lexical chain length. If $l_{w,d}$ is the lexical chain length (in sentences) for a term w in document d and L_d is the sentence-wise length of the document, then the term frequency of w is weighted by $l_{w,d}/L_d$.

Step 6: Classifier Training:

The documents output by step 4 are positively labeled documents. They are used to train a set of binary SVM classifiers, one category. All documents labeled with a given category are employed to train its classifier.

Step 7: Testing:

The test documents are applied to the trained classifiers to be finally assigned their appropriate categories.

V. DISCUSSION

The TC system proposed above differs from reported schemes with the following innovations. In this

A. Application of context for TC:

In this paper, context driven information is utilized for keyword generation as well as for document representation.

B. Keyword extraction:

We use a three pronged approach to extracting keywords automatically starting with only category names using,

- i. An LSA space vector similarity between documents and category names
- ii. Deriving lexical references to category names and
- iii. Utilizing the strong semantic connection between concepts in the vicinity of identified keywords.

While the first method has been tried out in [22] and the first two combined has been reported in [12], our algorithm augments both kinds of keywords by tapping distance based context. The combined approach for keyword extraction is geared towards an expanded and more cohesive keyword set.

Document to category matching:

In [12,22], the authors have used the cosine similarity metric to map documents with categories. This does not truly

capture their faithfulness to a given category. The r-score used here encodes the extent to which a document represents a category. It encapsulates both parameters ; the fraction of the keywords present in a document and the weighted average keywords frequency

Weighting TF-IDF with length of lexical chain:

To allow categorization for applications that require a vertical search based on a topic, we use lexical chaining that allows us to measure the extent of coverage on a concept. Being compute-intensive, we allow this feature as a user-requested option.

Tunable Guiding parameters:

Parameters such as maximum level of lexical referencing, vicinity score threshold and category assignment factor can be experimentally tuned. There is flexibility to adjust them for different applications and input categories.

VI. CONCLUSION

In conclusion, we have proposed a comprehensive scheme for context based TC that starts its job with only category names. A cohesive set of keywords is generated by exploiting three kinds of semantic cohesion; lexical references, implicit context and vicinity induced cohesion. Semantic cohesion is also utilized for representing documents as context oriented feature sets. While lexical references bring in generalization, an optional lexical chaining scheme allows the depth of coverage of concepts to influence the classification decision. The application of context at various levels gives a framework for more meaning-driven classification which cannot be expected from a purely bag-of-words approach.

REFERENCES

- [1] F. Sebastani, "Text categorization", *Trans. of State-of-the-art in Science and Engineering, Procs Text Mining and its Applications to Intelligence, CRM and Knowledge Management*, volume 17, 2005, WIT Press.
- [2] S. Pradhan *et.al.*, "Support vector learning for semantic argument classification," *Machine Learning*, 60, pp 11-39, 2005.
- [3] William W. Cohen and Y. Singer, "Context Sensitive Learning Methods for Text Categorization," *ACM Transaction on Information Systems*, Vol 17 No.2, Pages 141-173, 1999.
- [4] I. Androutsopoulos *et.al.*, "Learning to filter spam mail: a Bayes and a memory based approach," *Procs of the workshop "Machine Learning and Textual Information Access"*, 4th European Conference on Principles and Practice of Knowledge discovery in Databases, 2000.
- [5] Q. Wang *al.*, "SVM Based Spam Filter with Active and Online Learning", *Procs. of the TREC Conference*, 2006.
- [6] Jiang Su; Harry Zhang, "A Fast Decision Tree learning algorithm", *Procs of the 21st conf. on AI-Vol. 1 Boston*, Pages 500-505, , 2006.
- [7] Stephen Bloehdorn *et al.*, "Boosting for text classification with semantic features", *Procs. of the MSW 2004 workshop at the 10th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, AUG (2004), p. 70-87.
- [8] El-Sayed M. El-Alfy, Fares S. Al-Qunaieer, "A Fuzzy Similarity Approach for Automated Spam Filtering", *Procs. of the*

2008 IEEE/ACS International Conference on Computer Systems and Applications - Volume 00, Pages 544-550, 2008.

[9] P.I. Nakov, P.M. Dobrikov, "Non-Parametric Spam Filtering Based on KNN and LSA", *Procs of the 33th National Spring Conference*, 2004.

[10] Yiming Yang et al, "A comparative Study on Feature Selection in Text Classification", *Proceedings of ICML-97, 14th International Conference on Machine Learning*, page 412-420. Nashville, US, Morgan Kaufmann Publishers, San Francisco, US, (1997)

[11] Marco Erandes et al, "An Adaptive Context Based Algorithm for Term Weighting", *Proceedings of the 20th international joint conference on Artificial intelligence*, 2748-2753, 2007

[12] Libby Barak et al, "Text Categorization from Category Name via Lexical Reference", *Proc. of NAACL HLT 2009: Short Papers*, pages 33-36, June 2009.

[13] Diwakar Padmaraju et al, "Applying Lexical Semantics to Improve Text Classification", <http://web2py.iit.ac.in/publications/default/download/inproceedings.pdf.9ecb6867-0fb0-48a5-8020-0310468d3275.pdf>

[14] Reuters dataset: www.reuters.com

[15] OHSUMED test collection dataset: <http://medir.ohsu.edu/~hersh/sigir-94-ohsumed.pdf>

[16] FAODOC test collection dataset: www.tesisenxarxa.net/TESIS_UPC/AVAILABLE/TDX.../12ANNEX2.pdf

[17] Na Luo et al, "Using Co-training and semantic feature extraction for positive and unlabeled text classification", 2008 International Seminar on Future Information Technology and Management Engineering, <http://ieeexplore.ieee.org/stamp/stamp.jsp?arnumber=04746478>

[18] <http://www.brothersoft.com/downloads/nzb-senet.html>

[19] <http://WordNet.princeton.edu/WordNet/download/>

[20] http://en.wikipedia.org/wiki/Latent_semantic_analysis

[21] Halliday, M.A.K. and Hasan, R., *Cohesion In English*, Longman 1976.

[22] Gliozzo et al, "Investigating Unsupervised Learning for Text Categorization Botstrapping", *Proc. of Human Languages Technology Conference and Conference on Empirical Methods in Natural Languages*, Pages 129-136, October 2005

```

CTC(categories  $C=\{c_i\}$ , Unlabeled documents  $D=\{d_j\}$ , Lexical
referencing level  $k$ , Vicinity score threshold  $\tau_v$ , Stopwords  $\{sw\}$ ,
Category-assignment factor  $F$ , Lexical chain option  $l$ )
Resources: wordnet.
Begin {
  Step 1: Pre-process{
    1.1  $\forall d_j$  eliminate stop words:
         $\forall w_k$  in  $d_j$ , if  $w_k \in d_j \cap \{sw\}$  then  $d_j = d_j / w_k$ 
    1.2 Stemmer replaces inflected words with root forms
    1.3 Partition  $D$  into training set  $D_R$  and testing set  $D_E$ 
  }
  Step 2: Extract keywords{
     $\forall c_i$  {
      2.1 Create keywords of:
        -lexically related terms  $AK_{lex}(c_i)$ .
        - implicit context-related terms  $AK_{mc}(c_i)$  in LSA space
      2.2 Augment with vicinity connected keywords to make
         $AK_{lex}(c_i)$  and  $AK_{mc}(c_i)$ .
    }
  Step 3: Calculate similarity scores: {
     $\forall$  document  $d_j$ 
     $\forall$  category  $c_i$  {
      3.1 Lexical keywords similarity score  $= \rho(AK_{lex}(c_i), d_j)$ 
      3.2 LSA keywords similarity score  $= \rho(AK_{mc}(c_i), d_j)$ 
      3.3 Similarity score
         $\rho(c_i, d_j) = \rho(AK_{lex}(c_i), d_j) * \rho(AK_{mc}(c_i), d_j)$ 
    }
  }
  Step 4: Assign categories to documents:
     $\forall$  category  $c_i$  {
      4.1 Set  $cat(d_j) = \emptyset$ 
      4.2 Let maximum similarity score for category be:
         $\rho_{max}(c_i) = \max_j \{ \rho(c_i, d_j) \}$ 
      4.3 Assign  $c_i$  to  $d_b$  :  $d_b$  holds  $\rho_{max}(c_i)$ :  $cat(d_b) = cat(d_b) \cup c_i$ 
    }
      4.4  $\forall d_j$  if  $\rho(c_i, d_j) = F * \rho_{max}(c_i)$  then assign  $c_i$  to  $d_j$ :
         $cat(d_j) = cat(d_j) \cup c_i$ 
  Step 5: Represent documents as feature vectors:
    5.1  $\forall d_j$  tag terms with lexical references from wordnet.
    5.2  $\forall d_j$  Replace lexically related words with base word.
    5.3 If Lexical chain option  $L = 'Yes'$  then
       $\forall d_j$  Find lexical chain length  $l_{k,j}$  for each term  $w_{k,j}$  and
      calculate its weighted frequency  $= TF_{k,j} * l_{k,j} / L_i$ 
      else  $\forall d_j \forall w_{k,j}$  Find  $TF_{k,j}$ 
    5.4 Calculate TF-IDF of terms across all documents
  Step 6:  $\forall c_i$  Train a SVM classifier with  $\{d_j: c_i \in cat(d_j)\}$ 
  Step 7:  $\forall d_j$  Assign category to test documents by applying
    to each classifier.
} end

```

Figure 1: Pseudocode for the Context based Text Classifier